Victorian 6502 User Group Newsletter

KAOS

For People Who Have Got Smart

HARDWARE	DAVID ANEAR
SOFTWARE	JEFF RAE
FORTH	DAVID WILSON
AMATEUR RADIO	ROD DRYSDALE VK3BYU
EDUCATION	JEFF KERRY
LIBRARY	RON KERRY
TAPE LIBRARY	JOHN WHITEHEAD
DISK LIBRARY	WARREN SCHAECHE (B.H.)
NEWSLETTER	IAN EYLES
SYM	BRIAN CAMPBELL
SECRETARY	ROSEMARY EYLES

OSI SYM KIM AIM UK101 RABBLE 65

Registered by Australia Post
Publication No. VBG4212

ADDRESS ALL CORRESPONDENCE TO 10 FORBES ST., ESSENDON, VIC. 3040

Vol.4 No.1 October 1983

At the September meeting, our president, David Anear, announced that after three years it was time he stood down and gave someone else a chance to get involved with the running of the club. On behalf of all KAOS members we would like to thank David for all the work he has done for the club, especially for the three years he stood out in front and kept the meetings running smoothly.

On page one of last month's newsletter we wrote about the items Eric Lindsay has for sale but unfortunately we forgot to include Eric's address which is

Following on last month's item about overseas support for the OSI we have two more items. Bernie Wills has been in touch with The Program Exchange, 2920 West Moana, Reno, NV 89509 and they replied in less then two weeks, which is pretty good for a U.S. company. You may have seen their ads in Micro, they sell HEXDOS, HEXEDIT etc and a range of items for Aim, Sym, Kim and Apple as well as OSI.

The second item comes from Graham Gaiger and will be of interest to anyone using the Cegmon monitor. It is a 2 EPROM set called EXBASIC which gives 39 extra commands including: Labled GOTOs and GOSUBs, screen windows, REPEAT-UNTIL ELSE, DOKE and DEEK (double byte POKE and PEEK) and several DRAW commands. Another EPROM called EXBASIC2 which gives another 16 commands is also available. The EPROMs are 2716s and EXBASIC costs 18 Pound 50 and EXBASIC2 costs 8 Pound 50, both including postage. All the commands can be used in program or immediate mode and do not require prefixes, such as $ or &. One final point, information is supplied to enable you to insert your own BASIC commands. If you would like more information, KAOS can send you a copy of the information Graham sent us or you could contact the company direct. Their address is Vachette (Micros), Marshborough, Sandwich, Kent, England CT13 OPG.

The next meeting will be held on Sunday 30th October at 2pm at the Essendon Primary School Which is on the corner of Raleigh and Nicholson Streets, Essendon. Members will be welcome from 1pm.

The closing date for items for the next newsletter will be 11th November. Please note that as we will be posting the December newsletter early to avoid the Christmas rush, we would like items for inclusion before the 2nd December.

INDEX

64 CHARACTER VIDEO USING THE 6545 CRTC (Continued)

by Bernie Wills

Programming the 6545.

Firstly, some ideas about the screen display. For a 50 Hz frame rate, the frame time = 20,000 microseconds (uS). If a scan line takes 64 uS and each character is 8 scan lines in height, then number of character lines equals 20000/(64*8) , which = 39. If you squeeze in 82 character intervals in one scan line, time for 1 scan line = dot period * No. dots, which equals 82 * 8 /10/1,000,000 , or 65.6 uS for a 10 MHz dot clock. This would give a total of 38 character lines, calculated in the same way as above for the 64 uS scan period. The upshot of all this is that if you squeeze a couple more character intervals into a line so that 64 are just visible on your ex-TV, then you can get no more and no less than 38 lines of characters on your screen (in a timing sense). However, only 27 are visible in the configuration used here i.e. have a 27 x 64 screen, with memory map as for C4 video. (Excuse setting out of formulae BASIC style)

The comments above might help make sense of some of the programmable values used with the 6545.

An assembler listing of the initialisation program follows. For Dabug III, the diversion to this routine can be made at $FF0F JSR $FCA6. Alter this to JSR $E000 or where-ever you put the code, and include in the new code a JSR $FCA6 before executing RTS to $FF12. You may need to check this out fully as Dabug is not used with our boards. For Cegmon, divert to new code at $FF0A JSR $FE59 (screen clear), and include a screen clear in the new code. Assembler code shows Cegmon.

```
10    * = $E000
20    CRTC = $C400
30    CLOCK = $C200
40    ; WIDE SCREEN INITIALISATION SUBROUTINE
50    E000    A9 00      WIDLIN   LDA #0
60    E002    8D 00 C2            STA CLOCK
70    E005    A0 14               LDY #$14
80    E007    A2 00               LDX #0
90    E009    8A         WIDMOR   TXA
100   E00A    8D 00 C4            STA CRTC
110   E00D    BD 1B E0            LDA WIDTAB,X
120   E010    8D 01 C4            STA CRTC+1
130   E013    E8                  INX
140   E014    88                  DEY
150   E015    D0 F2               BNE WIDMOR
160   E017    20 59 FE            JSR $FE59 ; screen clr Cegmon
170   E01A    60                  RTS
180   E01B               WIDTAB   .BYTE $51,$40,$48,$52,$23,$0F,$20
190                               .BYTE $21,$10,$07
200   (ends $E02E)                .BYTE 0,0,0,0,0,0,0,0,0,0
```

The data loaded represent the following.

<u>Horizontal total -1</u> ($51) sets as dec. 82 the number of character time intervals in 1 trace across the screen. It works with any value in the range 79-82, but may vary with video monitor or TV in use. Setting it for 82 characters slightly compresses the characters, just allowing 64 on screen. I also run my TV from a variable d.c. supply, at about 11 volts, which also shrinks the screen a bit. As this value depends on the clock frequency, it means that you could use a surplus crystal with frequency above 10 MHz, and by appropriately adjusting values, achieve 64 displayed characters. So have a look in the junk box prior to paying up for a 10.0000000 MHz crystal.

Horizontal displayed ($40) is set for 64 characters. This is what keeps the display map the same as the C4, so this value must not be altered. The location $D000 will just be on the screen, as will $D03F at the end of the first line. In our displays, the first line for text starts at $D040, and the last starts at $D06C...use this info if setting Cegmon window.

Horizontal sync position ($48) is set to occur about half way between the end of the visible line and the end of the scan line. Can be varied a bit, with the effect of moving the display left or right.

Horiz/Vert sync widths ($52) read as two nibbles. Left nibble (5) sets vert sync pulse width equal to the time of 5 scan lines (about 300 microseconds), while the right nibble sets horiz sync pulse width equal to 2 character clock periods (about 2 microsec). Values not too critical.

Vertical total -1 ($23) sets the number of character lines as 36, a bit less than the calculated number for a 50 Hz screen. Not all lines appear on the screen. The value interacts with the next register.

Vertical total adjust ($0F) is the number of additional scan lines needed to complete a frame scan, and it acts as a fine tuning for frame time. It is not a very critical value, but is adjusted in conjunction with the vertical total. Increase V total by 1, decrease V adjust by 8, as there are 8 scan lines per character line. The combination of 36 character lines plus an adjust of dec. 15 gives about 38 character lines in total frame time, shown earlier to be necessary for a 50 Hz display.

We originally had circuits working with 60 Hz, to allow quick reversion to old video while sorting out this mess (which it was for a while). This is a good time to mention..PULL OUT OLD VIDEO 8T28's BEFORE USING NEW BOARD, this will isolate your old video circuitry.

Vertical displayed ($20) also works with other values, as the monitor restricts the screen display anyway, in our case to 27 lines in Basic, but characters can be Poked elsewhere. (and m/c

Vert sync position ($21) comes at character row 33. Used to adjust vertical position of the display.

Mode control ($10) is set for non-interlaced display (the only way the 6545-1 chip works, although Synertek list a 6545 which is supposed to have interlacing capability. This wasn't available when boards built) ; also set to delay the display enable signal from the chip by one character time. This delay (0 or 1) may need tuning for your system..see Synertek notes for full significance.

Scan lines -1 (7) set for an 8 line character as in OSI. To alter this, would need to reburn the character generator, and address it differently. (see info in Synertek data on raster address lines). The experimenter could develop a high resolution character set, with genuine descenders, on a larger matrix, but at the expense of the No. of character lines.

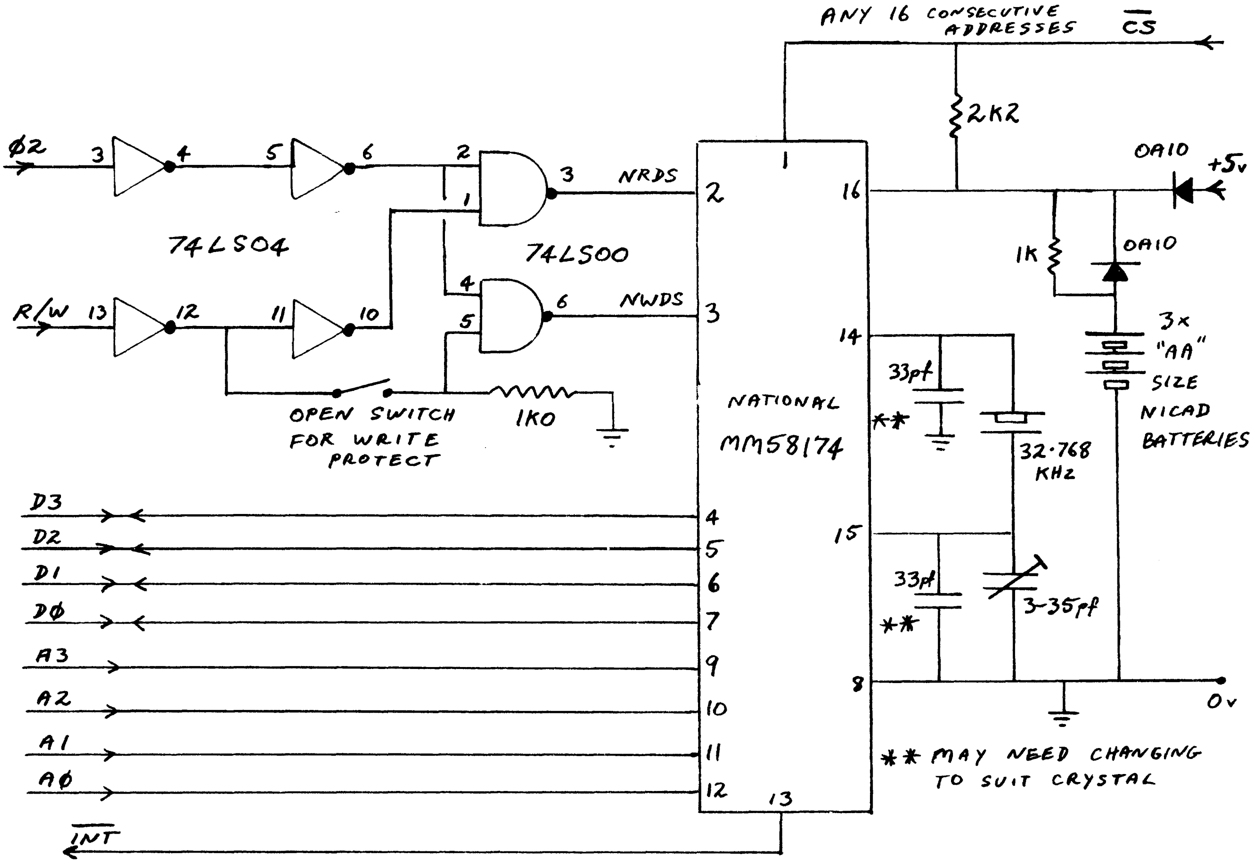
The other programmable registers are not relevant to the way the 6545 has been used here, and 0 is the value loaded. The corresponding values to those above for 24 x 24 are (hex) $28 20 25 52 23 0F 20 22 10 07 . To switch the dot clock to 5 MHz, change LDA #0 to LDA #1 in first program line of code.

Other character sets if used can be chosen by POKEing appropriate even numbers to an address in the range $C200-C3FF (64 chars), or odd numbers for 24 character screens. eg POKE 50000,0 is for default character generator and 64 screen width. We used a Basic program for fine tuning once the general display was working. Ignore all the zero's, and use a loop to do the same thing done by the m/c routine.

Superboard

October, 1983.
Newsletter of the Ohio Superboard User Group, 146 York Street, Nundah, 4012

Real Time CALENDAR CLOCK by Graham Gaiger.



This project was built on the proto area of the Rabble board. If you do not use a PIA or VIA to interface the clock chip, then the data lines D4 to D7 must be terminated with 1.5k resistors to 0 volts rail.

```
100 REM Rabble Calendar-Clock activation program by Graham Gaiger.
110 PRINT CHR$(26);"DISABLE WRITE PROTECT":PRINT:AD=49248
120 POKE AD,Ø:POKE AD+14,Ø:POKE AD+15,Ø:POKE 11,Ø:POKE 12,253
130 FOR R=4 TO 13:READ A$,A:PRINT"ENTER ";A$;:INPUT D:T=INT(D/1Ø)
140 U=D-T*1Ø:POKE AD+R,U:IF A=1 THEN R=R+A:POKE AD+R,T
150 NEXT R:PRINT:PRINT"ANY KEY STARTS CLOCK";:X=USR(X):PRINT
160 POKE AD+14,1:PRINT:PRINT"ENABLE WRITE PROTECT":END
170 DATA MINUTES,1,HOURS,1,DATE,1,DAY OF WEEK,Ø,MONTH,1,LEAP CODE,Ø
```

Notes:- Normally, Sunday is day 1 of the week but you choose.
Leap year code is 8 if this is a leap year,4 if last year was, 2 if it was two years ago, and 1 if next year is a leap year.

```
2ØØ REM Display Clock demo by Graham Gaiger, VK6ZGG
21Ø AD=49248:SC=54213:REM SC is screen location to suit C1P screen.
22Ø R=7:FOR L=Ø TO 7:D=PEEK(AD+R)+48:IFD=63 THEN 22Ø
23Ø POKE SC+L,D:R=R-1:IF R/2<>INT(R/2)AND R>2 THEN L=L+1:POKE SC+L,58
24Ø NEXT L:GOTO 22Ø
```

— SUPERBOARD —

PUZZLE - AND A CHALLENGE

If you like puzzles, then this short one will be of interest. It qualifies as one of the simplest looking, but meanest puzzles of all time. It should run as published here, on any variation of OSI computer. With very few changes, it should run on virtually any machine.

For those who wish to convert for another machine type, the POKEs in line 110 can be eliminated, and the USR(X) calls changed to a INKEY, GET, or INPUT statement wherever they appear. The PEEKs are part of this function.

```
1ØØ REM STAR SHOOT by Marty Vlekkert, 1979
11Ø PRINT CHR$(26)TAB(6)"STAR SHOOT":POKE 11,Ø:POKE 12,253
12Ø PRINT:PRINT"THE OBJECT OF THE GAME"
13Ø PRINT:PRINT"IS TO CHANGE THIS:-"
14Ø PRINT:PRINT":::":PRINT":*:":PRINT":::"
15Ø PRINT:PRINT"TO THIS:-":PRINT:PRINT"***":PRINT"*:*":PRINT"***"
16Ø PRINT:PRINT"IT'S HARDER THAN IT"
17Ø PRINT:PRINT"LOOKS WITH A MINIMUM OF":PRINT:PRINT"11 MOVES."
18Ø PRINT:PRINT"INSTRUCTIONS ?";:X=USR(X):IF PEEK(531)=89 GOTO 32Ø
19Ø PRINT:FOR R=1 TO 9:A(R)=-1:NEXT:A(5)=1:T=Ø:GOSUB 4ØØ
2ØØ PRINT"WHERE TO SHOOT '1-9'?";:X=USR(X):B=PEEK(531)-48:PRINT B
21Ø IF B<1 OR B>9 GOTO 2ØØ
22Ø IFA(B)=-1 THEN PRINT B;"IS NOT A STAR":PRINT:GOTO 2ØØ
23Ø READ C:IF B=C GOTO 25Ø
24Ø FOR R=1 TO 4:READ C:NEXT:GOTO 23Ø
25Ø FOR R=1 TO 5:A(C)=-A(C):READ C:NEXT
26Ø DATA 1,2,4,5,Ø,2,1,3,Ø,Ø,3,2,5,6,Ø,4,1,7,Ø,Ø,5,2,4
27Ø DATA 6,8,6,3,9,Ø,Ø,7,4,5,8,Ø,8,7,9,Ø,Ø,9,6,5,8,Ø,Ø
28Ø T=T+1:S=Ø:FOR R=1 TO 9:S=A(R)+S:NEXT
29Ø IF S=-9 THEN PRINT:PRINT"NO STARS LEFT!":PRINT:GOTO 44Ø
3ØØ IF S=7 AND A(5)=-1 THEN PRINT"YOU GOT IT IN";T;"TRIES":GOTO 44Ø
31Ø GOSUB 4ØØ:GOTO  2ØØ
32Ø PRINT:PRINT"CENTRE SHOT, 5":PRINT:PRINT":::"TAB(12)":*:"
33Ø PRINT":*: BECOMES *:*":PRINT":::"TAB(12)":*:"
34Ø PRINT:PRINT"SIDE SHOT, 2 (AND 4,6,8)":PRINT":*:"TAB(12)"*:*"
35Ø PRINT"::: BECOMES :::":PRINT":::"TAB(12)":::"
36Ø PRINT:PRINT"CORNER SHOT, 1 (3,7,9)":PRINT:PRINT"*::"TAB(12)":*:"
37Ø PRINT"::: BECOMES **:":PRINT":::"TAB(12)":::"
38Ø PRINT:PRINT"HERE ARE THE NUMBERS:-":GOSUB 43Ø
39Ø PRINT"READY ?";:X=USR(X):PRINT:GOTO 19Ø
4ØØ FOR R=Ø TO 2:FOR J=1 TO 3:IF A(R*3+J)=-1 THEN PRINT":";:GOTO 42Ø
41Ø PRINT"*";
42Ø NEXT:PRINT:NEXT:PRINT:RESTORE
43Ø PRINT:PRINT"123":PRINT"456":PRINT"789":PRINT:RETURN
44Ø PRINT:IF T<>11 THEN PRINT"YOU CAN DO BETTER":PRINT:GOTO 39Ø
```

Now for the challenge!

Write a program to solve the problem, and give one of the eleven move answers. The first working one to reach me by surface mail, on a cassette, recorded mono and at 3ØØ Baud in OSI format, will win a $5 cash prize!

Please indicate on the tape the approximate time the program takes to solve the problem on the OSI using a 1MHz CPU clock. Should 2 or more tapes reach me on the one day, the most time efficient one will win. This offer is open to all KAOS members and closes with publication of the next KAOS. Your cassette will be returned to you. I reserve the right to publish any entry in Superboard under the Author's name.   Go to it!

— SUPERBOARD —

SOFTWARE REVIEW - Super Defender.

Super Defender is a M/C arcade game occupying $Ø4ØØ to $1C76. It is based on the Defender arcade game. Many arcade game enthusiasts consider that Defender is the finest arcade game yet written, though Pacman games have been the most popular money earners.

In Super Defender, you patrol in your rocket over the surface of a small planet in order to defend your humans. Aliens materialise in space and try to abduct the humans. You control the direction ,speed, and firepower of your rocket with 6 keys, 4 being in constant use. Your main weapon is a forward firing laser, and you have a limited supply of "smart bombs".

Your task is to shoot the aliens, particularly those who are carrying off your humans. Some skilful flying is required to catch the humans, and transport them back down to the planet surface. If they fall back, they are destroyed if the distance of fall is too great.

The aliens are not defenceless, and will shoot at you with good accuracy. Setting off a smart bomb will destroy all aliens on the screen. This will deplete your limited stock of smart bombs, but sometimes it is the only way to save humans when several are being carried off together. If you fire at an alien, but hit a human, the human only will be destroyed.

It is important to prevent human carrying aliens from reaching the top of the screen. If they do, the human dies and the alien turns into a mutant, which is more aggressive than a normal alien. If all humans on the planet are destroyed, the planet explodes, all remaining aliens turn into mutants, and the speed of the mutants increases.

You score points for each alien destroyed, and even more if you save humans. The score, and the number of remaining smart bombs and rockets, are displayed on the top left of the screen. At top centre is your long range scanner, where you can see what is happening on the entire surface of the planet. Where a human is being abducted, he has his arm extended. You get a bonus rocket and smart bomb for every 10,000 points you score. When all your rockets have been destroyed, the game ends.

This game is extremely fast moving, and good reflexes and skills are required to achieve a high score. It is easily the very best arcade game that I have ever played on the OSI, and is positively addictive.

Super Defender is available from D.M.P. Systems, 319 Hampton Blvd., Rochester, N.Y. 14612, U.S.A. , at a cost of US$14.95 + US$2.00 postage. It can be ordered to suit C1P or C4P. The OSUG library has a C1P version.

- -

BASIC 4 - Bits and Pieces

There is a very simple way to Auto-Run machine code programs, selectable when you load. Just POKE the starting address of the program into locations 1 and 2. eg:- Starting address is $Ø235. POKE 1,53:POKE 2,2 (RETURN)
Now load in the normal way, not enclosing the inverted commas. Another advantage of this procedure is that if you hit BREAK, then W will restart the program.
If you interrupt the save mode with a BREAK, then attempt to list your Basic program, it won't be there. Simply type X=USR(X) to reset the pointers back to the start of the program, and then you can run or save it.

Ed Richardson.

NEXT MONTH :- Finding memory faults on the OSI. Review on an amazing game. A mini adventure for you to type in.

THE MEETING WAS KAOS
by King Corky

Well, I am back after a fabulous 4 weeks in 'The States'. Unfortunately the speed we were travelling and the coincidence of arriving at the right place at the wrong time, prevented me from learning much more about the American computing scene than we already know.

The only OSI software I saw we have already, and the only hardware I saw was either ridiculously expensive or just out of the range of our budget. For example, a 48 pin 470 board, modified as an I/O board, was being offered for $280. This board had nothing more on it than a couple of chips, a PIA and a parallel connector and ribbon. In marked contrast, at the same shop, I was offered a 256K C3, complete with 8" floppy, set-up for 6 terminal operation, plus HARD DISK. Price $2500.

Anyway back to the meeting, which by the way was our Annual General Meeting. David Anear, after three years, stepped down from the presidential podium and handed the rod of office to the keeper of our disk library, Warren Schaeche. On behalf of all members, I would like to echo Tony Durrant's words of praise and thanks to David for the Magnificent job done over the last three years. Without people like D.J.A., Ian and Rosemary Eyles, this club would be nothing and to them I say thanks. Also on behalf of KAOS members I congratulate Warren on his winning the chair, totally un-opposed and unanimously agreed to.

Ray Gardiner gave us the latest update on the Rabble system and it's progressing in leaps and bounds, (even if small ones). George has a couple of early Rabble boards for sale cheap. The stand-alone Hi-Res board is not too far away, and not too expensive, (can't wait). The October meeting will be an early start for those of us who are prepared to dismember, lug, cart and re-assemble their systems and hope they work, 9.30 am at the school. If you do not normally attend these few-per-year type sessions, let me tell you it's a great day. A lot of fun can be had teaching the kids basic programming skills and it's also a great time for comparing programs. Please let Rosemary know if you can come early.

SOFTWARE for HI RES GRAPHICS
by Bernie Wills

After recent homework, I am able to supply on cassette, software to use the character set sold to KAOS members. The chips were originally supplied with a listing which required some effort to convert for other screen formats or ways of software selecting character sets. I can assemble the required machine code and patch to BASIC for the following variations:
Dabug / Synmon / Cegmon monitors, 24 x 24 or 32 x 64 (48) screens, 7474 latch at $D800 or elseware, 74174 latch anywhere. (C1 KEYBOARD ONLY).

Those who have purchased the chip can obtain the code by forwarding a C60 tape (no case), prefeıably in a jiffy bag, with a 50c stamp to cover postage. You will need to specify the factors above, in particular the methoed of software selection and POKE values if using the 74174. Programs can later be easily converted from 7474 to 74174 switching owing to the way the m/c has been written. On request with a cassette, a sheet of notes on how to use the 74174 will be provided, as this is not included in the original documentation. (It was described to several buyers who asked).

New purchasers of the sets should include a cassette with order, and the program will be sent at no extra cost. Details of proposed installation (screen etc) will still be needed.

Enquiries to B. Wills,

EPROMs
By John Whitehead

Some club members have had problems programing EPROMs using my hardware and software because they were using the wrong type of EPROM.

I built my programer using switches to select the three types I use. These are marked in the chart with a *. None of the other types can be programed on my setup without it being modified, but a 2732A can be read when switched to 2732.

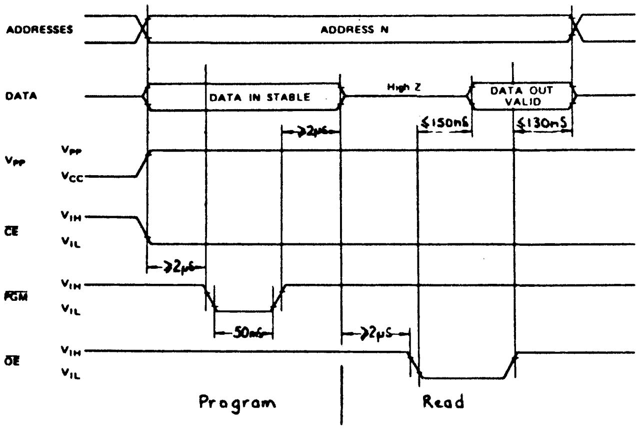
If you want to program other types, the circuit could be altered by replacing the switches with a DIP socket and a plug for each type.

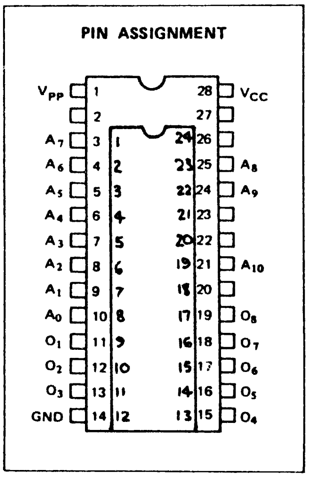
EPROMs are MOS devices and should be protected from static, they are erased with a 2537 Angstrom light of 12mW/cm2 for 15 to 20 minutes and the window should be covered with an opaque label to prevent slow erasure from fluorescent or sunlight.

The chart below shows variations between types. If anyone has any further data, please let me know.
A = address. OE = output enable.
NC = not connected internally, can be +5V.
PGM = TTL level program pulse, 55ms plus or minus 5. PGM is either a positve or negative going pulse as shown. Where no pulse type is marked in the chart, I dont know it.
CS or CE = chip select. 0 = TTL level zero. 1 = TTL level one.
Vpp = program power, to be within 0.5V of value shown, including overshoot. Vpp must not be on before Vcc.

TMS 2716 Texas	2716 or Texas TMS 2516 *			27C16			2532			2732 *			2732A			24 pin type pin	28 pin type pin	2564 Texas			2764 Intel Hitachi Fujitsu *			2764 Mostek		
FuncReadProg	Func	Read	Prog	Func	Read	Prog	Func	Read	Prog	Func	Read	Prog	Func	Read	Prog			Func	Read	Prog	Func	Read	Prog	Func	Read	Prog
Needs +12V &																	1	Vpp			Vpp	+5	+21			
-5V supply																	2	oV			A12			A12		
	$\overline{CS}$	0	⎍	$\overline{CS}$	0		A11			$\overline{CS}$	0	⊔	$\overline{CS}$	0	⊔	18	20	A11			$\overline{CE}$	0	0	$\overline{CS}$	0	PGM
	$\overline{OE}$	0	1	$\overline{OE}$	0	1	$\overline{CS}$	0	⊔	$\overline{OE}$	0	+25	$\overline{OE}$	0	+21	20	22			PGM	$\overline{OE}$	0	1	Vpp		
	Vpp	+5	+25	Vpp	+5		Vpp	+5	+25	A11			A11			21	23	A12			A11			A11		
	Vcc	+5	+5	Vcc	+5	+5	Vcc	+5	+5	Vcc	+5	+5	Vcc	+5	+5	24	26				NC					
																	27		0		PGM	1	⊔			

2764 EPROM





THE BEGINNING MACHINE LANGUAGE PROGRAMMER Part 16

by David Dodds

In OSI computers the 6502 does not actually deal with the tape recorder at the port. A device known as an Asynchronous Communications Interface Adaptor or ACIA provides a sort of letter box where the 6502 can post or pick up information, character by character. (Byte by byte if you prefer.)

Normally the standard Input/Output routines resident in the computer handle the task of dealing with the ACIA but these routines are configured to only handle data values up to $7F. For ARTIST to function effectively all values between 0--$FF must be supported. This limitation lead to the development of a set of reliable high speed load/save routines which utilize more of the power of the ACIA than the OSI routines use.

The ACIA has the task of converting 8 bit parallel data to a serial data stream or vice versa. During sending operation error checking information is added to the data. During a receive operation the information is stripped out. The ACIA may be configured for a variety of data formats and also provides status and error reporting facilities.

To the 6502 (and the programmer) the ACIA appears to be two consecutive memory locations but actually consists of 2 write only registers and 2 read only registers. One location provides the 'letter box' where data is sent or received. The second location in write operation configures the operation of the ACIA; in read operations, information or errors and the readiness of data is available.

The registers in the ACIA are known as:

1. Received Data Register (Read Only)
2. Transmit Data Register (Write Only)
3. Status Register (Read Only)
4. Control Register (Write Only)

The Received and Transmit Data Registers are used only for the transfer of data. The Status Register provides a variety of useful information about the operation of the ACIA including:-

Receive Data Register Full (Incoming data is ready)
Transmit Data Register Empty (Ready to send more data)
Parity Error (Checksum error)

The information is in true/false form as 1 bit of the Status Register.

Some of the functions of the ACIA which can be user determined through the Control Register are:-

Data word length (7 or 8 bits)
Parity Checking (a kind of mini checksum)
Master Reset (Clear the ACIA for use)
Check divide rate.

The routines in ARTIST configure the ACIA to provide

* 8 bit data word
* parity checking so that the parity bit indicates whether the sum of bits in the data is an even number
* Divide by 16 clocks (The usual value at 300 baud)

To achieve this, the first step is to perform a master reset by storing the value 3 at the ACIA, then the control register is loaded with the value %00011001 or $49 to set up the desired facilities.

The procedure is simply

```
1  LDA  #3
2  STA  ACIA
3  LDA  #%00011001
4  STA  ACIA
```

Changing the value of line 3 alters the operation of the ACIA. In ARTIST the send/receive routines make use of 2 instructions which have not yet been covered. These are the logic function AND one known as Bit Test or BIT:

The instruction AND causes the contents of the Accummulator to be logically ANDed with the memory location (or data) specified in the operand of the instruction. The result is placed in the Accummulator where corresponding bits of the two values are both 1s the resulting bit in the Accummulator is set to 1. Otherwise it will be zero.
For example. A contains $65 = %01100101
after execution of AND #$0F = %00001111
result in A is $05 = %00000101

The N and Z flags in the Status Registers will be set according to the result of the AND. In the example the result is positive and non-zero so that N = 0 and Z = 0.

The BIT instruction also performs a logical AND between the contents of the Accumulator and memory but neither of the values involved is altered as only the Status flags are affected. The Z flag reflects the result of the AND while the N and V flags receive the actual value contained in bits 6 and 7 (the 7th and 8th or Most Significent Bits) of the memory location addressed by the BIT.

For example if A contains 2 %00000010
and location $F000 contains $82 %10000010
then BIT $F000 will yield A = 2
$F000 = $82
Z = 0
N = 1 (bit 7 of $F000)
V = 0 (bit 6 of $F000)

Next month we will show how to use these instruction to interogate the Status Registers of the ACIA.

BIG SALE AT COMP-SOFT

Special offers to KAOS members from COMP-SOFT

TASAN Video Boards, bare	$25.00	inc. tax
ALPHA-80 Printers	$450.00	" "
MPI B51 Disk Drives	$250.00	" "
AMDEK Green Screens	$265.00	" "
Box of printer paper (2000 sheets)	$30.00	" "
Hitachi PEACH computer, 48K RAM, 2 Disk Drives Disk Controller & lots of Software	$2200.00	" "

SOURCE of COMPDOS 1.3 released.

Owners of 1.3 may now aquire the Source	$15.00	" "
COMPDOS 1.3	$35.00	" "
COMPDOS 1.3 plus Source	$50.00	" "

RABBLE SYSTEM .... $1600.00 " "

As a special introductory offer we have 10 only Rabble 65 computers at this very low price. They are fully assembled and tested and in a case with a power supply and 2 x 8" look-a-like disk drives. The Hi-Tek keyboard is in a separate case and has a numeric keypad and 18 function keys. All cables and manuals will be supplied and the 10 lucky buyers can also negotiate a special deal on screens and printers.

COMP-SOFT 235 Swan St, Richmond 3121, ph 03 428 5269 or 03 429 9686

MY SUPERBOARD Series II. Part 7

by John Whitehead

Some months ago I saw an article written by Don VanSychel in the Aardvark journal, which gave a routine to correctly decode the C4 keyboard.

I typed this new routine into the assembler and tried it out. With the Shift lock up all worked correctly, but with it down, lower case was not available which is little improvement for BASIC. Another thing I did not like was the use of the Repeat key to get the characters that are obtained with Shift K L M N O & P. The Repeat key in the original routine is undecoded which means it's being pressed can be detected with POKEs and PEEKs etc without it altering the value of any other key pressed.

The original keyboard routine is in the 2k Monitor EPROM starting at $FD00 to $FDFF and in a C1P has extra code at $FCBE to $FCD4 to convert the data to suit the C1P hardware.

I altered the new keyboard routine to make lower case available with Shift lock down or up, but tò retain all control codes and use the CTRL key in place of Repeat I needed more room. After some hints from David Anear, I was able to obtain more room by reducing and improving the OSI 48x12 screen driver. The result of all this is DABUG 3J which is in a 2k 2716 EPROM and is a direct replacment for DABUG 3.

The differences between this and DABUG 3 are :-

1. Correctly decoded keyboard.
   With the Shift lock down it gives upper case as normal. When left or right Shift is down it gives lower case for use with BASIC or the Assembler. This makes adding lower case text easy.

   With the Shift lock up, keys A to Z are inverted in that it gives lower case when the left or right Shift is up and upper case when it is down like a normal typewriter.

   The numerical keys are not affected by the Shift lock and give correct characters as marked.

   To get the characters that were originally obtained with Shift K L M N and O, non alfa keys are used with the CTRL key as below. The Hex value is in a table so that it can be customised if required before burning the EPROM.

   The backspace key is now CTRL RUBOUT for convenience and RUBOUT remains as Screenclear.

   All non printing control codes from $01 to $1F are obtainable and CTRL A to Z are not affected by Shift or Shift lock.

   The REPEAT key remains undecoded for special use.

   The keyboard routine is entered at $FD00 and puts the ASCII value of the key pressed into the accumulator. It does not exit until a key has been pressed. The X and Y registers have the same value on exit as on entry. A subroutine at $FD21 will put the ASCII value of a keypress in the accumulator and if no key is pressed will put zero in the accumulator and then return.

CTRL key.	Hex.	character.	CTRL key.	Hex.	character.
,	EC	cursor left	5	5C	backslash
-	1E		6	EF	original /
.	EE	cursor right	7	5E	power arrow
/	FC	cursor down	8	5B	square bracket
0	40	at sign	9	5D	square bracket
1	1D		:	1C	
2	1F		;	FB	cursor up
3	7B	curly bracket	RUB OUT	5F	backspace
4	7C	curly bracket			

2. BASIC warm start OM error fixed.

3. E added to the BREAK selection to enter EXMON in EPROM in the 48x12 mode and Shift lock does not need to be down after a BREAK.
(Exmon in EPROM is not needed for any of the other functions to work)

4. The 48x12 screen driver has been altered to fix the faulty cursor character down the left side of the screen when viewing a BASIC tape or using the Assembler. All characters from 1 to 126 can be PRINTed. Also one more non scroll line is available at the bottom of the screen without loss of screen size for poking the time, line number tracing or soft front panel.

Dabug screen editing, single key basic and the 65V monitor are unaltered except that cursor up & down is changed to CTRL ; & / but can be left as U & D.
The entry points of all routines are the same as DABUG 3, so all programs that use complete routines are fully compatable. WP6502 has it's own keyboard routine so the keyboard works as before. The graphics adventure game CASTLE uses part of the original keyboard routine so will not work. I have modified Castle to work with DABUG 3 or 3J but it needs 16k to run because all lines now have numbers. After having DABUG 3J in my SUPERBOARD for 4 months, I have not found any other programs that use part of the keyboard routine, and if I do I will fix them.

The list of entry points to routines in DABUG 3 or 3J.

F800	Dabug input	FAA1	Turn on Dabug
FB00	Dabug output routine leads to 48x12 driver or original routine at $FF69	FC00	Disc
		FCA6	ACIA reset
		FCD5	Screen clear
FD00	Keyboard	FE00	Monitor
FE43	used by checksum loader	FE80	ACIA
FEED	JMP $FD00	FF00	Reset. Here after BREAK
FF29	used by BASIC	FF35	used by Space invasion
FF69	Normal output and save	FF8B	Load vector points here
FF96	Save vector points here	FF9B	CTRL C vector points here
FFBA	Normal input and load	FFE0	Data used by 24x24 screen driver in BASIC ROM
FFEB	to FFFF vector addresses		

New entry point in DABUG 3J is $FD21 Get key

DABUG 3J is available from myself for $10 plus $1 p & p.
John Whitehead.

WORD PROCESSOR (LINE EDITOR)
by Rod Drysdale VK3BYU

I find this program useful when writing letters or memos, due to it's simplicity.

There are a number of improvements that could be made to the program, such as centering headings etc, but once it was running I got too lazy to include them.

Note: When leading spaces are required you must enter a quotation mark at the start of the line.

```
10 DIMA$(200)
20 REM INPUT DATA
30 LN=0:A=0
40 GOSUB 1010
50 PRINT"                    INPUT TEXT"
60 FOR K=1 TO 5:PRINT:NEXT K
70 IFA<10THENPRINT" ";A;:GOTO90
80 PRINT;A;
90 INPUTA$(A)
100 IFA$(A)="."THENA=A-1:GOTO120
110 A=A+1:GOTO70
120 PRINT: INPUT"TYPE IN COMMAND";ANS$
130 IF ANS$="A"THEN 240
140 IF ANS$="E"THEN GOTO  1030
150 IF ANS$="F"THEN GOSUB  310
160 IF ANS$="H"THEN GOSUB  400
170 IF ANS$="I"THEN GOSUB  530
180 IF ANS$="K"THEN GOSUB  610
190 IF ANS$="L"THEN GOSUB  670
200 IF ANS$="P"THEN GOSUB  750
210 IF ANS$="R"THEN GOSUB  280
220 IF ANS$="S"THEN GOSUB  880
230 GOTO 120
240 REM****** CONTINUE - APPEND ********
250 GOSUB 1010
260 A=A+1:GOTO 70
270 REM ***** REPLACE ******
280 INPUT"TYPE IN LINE NUMBER";T:GOSUB 1020
290 PRINT;T;: INPUTA$(T):RETURN
300 REM ***** FIND OR SEARCH ******
310 INPUT"TYPE SEARCH STRING"; ST$
320 GOSUB 1020
330 FORT=0TOA
340 L1=LEN(A$(T))
350 FOR R=1TOL1
360 L=LEN(ST$)
370 IFMID$(A$(T),R,L)=ST$THENPRINT;T;A$(T):RETURN
380 NEXTR:NEXTT
390 RETURN
400 REM********* HELP *********
410 GOSUB 1020
420 PRINT"A Append or continue"
430 PRINT"E End - return to Basis"
440 PRINT"F Find or search"
450 PRINT"H Help-prints this table"
460 PRINT"I Insert lines of text"
470 PRINT"K Kills lines by number"
480 PRINT"L Lists text"
490 PRINT"R Replaces lines of text"
500 PRINT"P Prints text on printer"
510 PRINT"S Substitutes text-global edit"
520 RETURN
530 REM**** INSERT TEXT ********
540 GOSUB 1020:INPUT"TYPE IN LINE NUMBER";T
550 IFT>A+1THEN PRINT"NO.TOO HIGH MUST BE LESS
     THAN";A+1:GOTO 540
560 A=A+1
570 FOR R=ATOT STEP-1:A$(R+1)=A$(R):NEXTR
580 PRINT;T;:INPUTA$(T)
590 IFA$(T)="."THENGOSUB 630:RETURN
600 T=T+1:GOTO 560
610 REM ****** KILL LINE ******
620 INPUT"LINE NUMBER";T
630 A=A-1
640 IF A<T THEN RETURN
650 FOR R=T TO A:A$(R)=A$(R+1):NEXTR
660 RETURN
670 REM**** LIST TEXT *****
680 PRINTCHR$(127)
690 SL=17
700 FOR R=0TOA
710 IFR>SLTHENGOSUB 1040:SL=SL+17
720 IF R<10THENPRINT" ";R;A$(R):GOTO 740
730 PRINT;R;A$(R)
740 NEXTR:PRINT:PRINT:PRINT:RETURN
750 REM**** PRINTER *******
760 PRINTCHR$(127):INPUT"Lm";LM
770 INPUT"Li ";LI
780 POKE517,1
790 FORP=1TO5:PRINT:NEXTP
800 FORR=0TOA
810 PRINTTAB(LM)A$(R)
820 IFR+5>LITHENGOSUB 990:LI=LI*2
830 NEXTR
840 GOSUB 990
850 POKE517,0
860 RETURN
870 REM**** SUBSTITUTE TEXT *****
880 GOSUB 1020
890 INPUT"SUB FROM";ST$:PRINT
900 INPUT"TO";X$:PRINT
910 GOSUB 330:PRINT
920 NW$=MID$(A$(T),1,R-1)
930 ED$=MID$(A$(T),R+L)
940 SS$=NW$+X$+ED$
950 GOSUB 1020:PRINT;T;SS$:GOSUB 1020
960 INPUT"IS THIS OK";Y$
970 IF Y$="Y"THENA$(T)=SS$
980 GOSUB 1020:RETURN
990 PRINTCHR$(12)
1000 RETURN
1010 PRINTCHR$(127):RETURN
1020 PRINT:PRINT:RETURN
1030 END
1040 REM****** WAIT FOR SPACE BAR *********
1050 POKE 530,1:K=57088
1060 Q=255
1070 POKE K,253
1080 IF PEEK(K)<>Q THEN POKE 530,0:RETURN
1090 GOTO 1070
```

KAOS WA

Our last meeting was again characterised by the addition of 3 new members.

As promised Lindsay Duus had his C4P with 8" drive working but unfortunately the person who was to bring some software did not arrive. So hopefully at our next meeting we will be able to see the system and see it running.

Our next Bi-monthly meeting is on Sunday Nov 20th at the address below (at 2.00pm). This is only a temporary arrangement. Our following meeting on 15th Jan will be back at our usual meeting place.

See you on Nov 20th. Gerry Ligermoet,

```
0 TOKEN CHECKSUM LOAD/SAVE by John Whitehead
1 This program is one of three I use to make copies of the clubs
2 machine code programs. It was written by Alan Cashin of Queensland.
3 At 300 baud it will load 3 times faster than OSI checksum format.
4 I have modified it so that I can put directions on a tape to save me
5 writing little notes. It also allows 1 line of POKEs which can be
6 used to restrict BASIC to below the m/code program.
7 It can save between 1 and 4 blocks of code of any length and
8 auto start the program.
9 To return to BASIC after loading answer $0000 to the Go at address.
10 :
30 REMARKABLE TOKEN LOAD by Alan Cashin. Mods by John Whitehead
35 POKE517,0:DIMD(3,3):AC=61440:FF=65535
40 PRINTCHR$(127)"Token checksum LOAD/SAVE":GOTO80
45 WAITAC,2:POKEAC+1,E:RETURN
50 FORA=0TO200:PRINTCHR$(0);:NEXT:PRINT:RETURN
55 DATAStart address ,End   address ,Go at or More
60 DATA.0013/95*5B*2C*A2*C8*AD*00*F0*4A*90*FA*AD*01*F0*E8*30*EF*.0016G
65 DATAA206814F18755395539002F654A901CACAD0F1A2FC202901E830FAE004F00B
70 DATAB557F0F5E00290CD6CFCFF85FBA2F74C18002901E9FFC1F660130655265690
75 DATA02E6552A90F5C93A900269069DA9CF6018002901FFFF57FFA9X
80 PRINT" get tape ready.":INPUT"Enter Program name ";P$
85 PRINT"Input 4 single line messages, a space = none
90 INPUTC$,D$,E$,F$
95 PRINT"Enter a line of commands begining with a ";CHR$(34)
100 INPUTG$
105 RESTORE:FORA=1TO3:READB$
110 PRINTB$;:INPUTX$:IFLEN(X$)=4THEN125
115 IFX$="M"ANDA=3THENE=24:GOTO145
120 PRINT"Not 4 hex":GOTO110
125 E=0:FORC=1TO4:F=ASC(MID$(X$,C,1)):IFF>47ANDF<58THENF=F-48:GOTO140
130 IFF>64ANDF<71THENF=F-55:GOTO140
135 GOTO120
140 E=E*16+F:NEXTC
145 D(N,A)=E:NEXTA:IFD(N,1)>D(N,2)THENPRINT"START > END":GOTO105
150 C=0:FORA=D(N,1)TOD(N,2):C=C-PEEK(A):IFC<0THENC=C+FF+1
155 NEXTA:D(N,0)=D(N,3):D(N,3)=C:D(N,2)=FF-D(N,2)+D(N,1)
160 IFD(N,0)=24THENN=N+1:GOTO105
165 PRINT" Start tape now":SAVE:GOSUB50
170 PRINT" 20 PRINTCHR$(127)"CHR$(34);P$
175 PRINT" 21 ";G$
180 PRINT" 22 PRINT"CHR$(34)"Token Checksum load"CHR$(34)"
185 PRINT" 23 PRINT:PRINT:PRINT:PRINT"CHR$(34)C$
190 PRINT" 24 PRINT"CHR$(34)D$
195 PRINT" 25 PRINT"CHR$(34)E$
200 PRINT" 26 PRINT"CHR$(34)F$
205 PRINT" 27 POKE11,67:POKE12,254:X=USR(X):END
210 PRINT"POKE515,0:RUN20";CHR$(13):GOSUB50
215 READB$:FORA=1TOLEN(B$):E=ASC(MID$(B$,A,1)):IFE=42THENE=13
220 GOSUB45:NEXTA:C=500:F=C
225 C=C+1:IFC>LEN(B$)THENREADB$:C=1
230 E=F:F=ASC(MID$(B$,C,1))-48:IFF>9THENF=F-7:IFF>15GOTO245
235 IFE<17THENE=E*16+F:GOSUB45:F=500
240 GOTO225
245 FORA=0TON:FORC=0TO3:F=INT(D(A,C)/256):E=D(A,C)-F*256:GOSUB45:E=F
250 GOSUB45:NEXTC:FORC=D(A,1)TOFF-D(A,2)+D(A,1):E=PEEK(C):GOSUB45
255 NEXTC,A:END
```

FOR SALE

MEMOREX communications terminal with speeds from 100 to 1200 baud has inbuilt cassette drive for taping of data to be transmitted or recording of received data. Printer has full 15 inch wide carriage, unit is in very good condition, just been serviced by manufacturer, ideal as a mean of signing onto mainframes, or as a microcomputer printer. Just as good as a ASR33 TTY. Price $200 O.N.O.
Chris Robertson

Superboard I in case with 16K memory. Full documentation and loads of programs including Pascelf and Assembler. $300 ONO
Martin Coull

Superboard case, large white fiberglass with metal base. Space for tape recorder, power supply and expansion boards. $30
Contact Peter Westley

OSI 600 and 610, 610 and Tasan video boards (32K of RAM) in metal case with 10 amp power supply and MPI B51 disk drive, has been modified to emulate a C4P. Complete with both serial and centronics printer ports. Complete set of manuals for DOS, Assembler and Extended Monitor etc. software included. COMPDOS 1.2, 1.3, PicoDOS, WP6502 and various games and utilities on disk and also 20 AARDVARK cassettes including 3 adventures, Chess, Gamon, Startrek and others. Price $650 O.N.O.
Chris Robertson

Tasker Expansion Mother board - including eight edge connectors and all ICs. $20
Tasker I/O board fully socketed (no ICs). $10
Digital cassette drive - 2 DC drive motors. Motorized head engage and disengage - info and circuits included. (no electronics) $25
Amateur T.V. converter for 426 MHz in diecast box and PL259 connectors. $25
Amateur T.V. converter for 579.25 MHz (repeater) in box including supply. $15
Contact Rod

C-3 OEM triple processor, Z80, 6502 and 6800 software selectable, with twin 8" disk drives and a Terminet 1200 hard copy terminal and null modem connection cable. Over 75 diskettes contain the following software:

* CPM Operating system with BASIC, FORTRAN, COBAL and MACRO ASSEMBLER languages and Escort disk.
* OS-65U, OS-65D and OS-65D Real Time Monitor operating systems with BASIC language, BASIC programs and the OSI 6502 assembler, C3 Utilities and Demos.
* OSI FORTH language system.
* USCD PASCAL system with PASCAL integer and floating point compilers, the FORTRAN 77 compiler and utilities.
* OS-DMS Inventory II, Accounts Payable and Accounts Receivable.
* OS Word Processor.

The computer comes with a complete set of books and manuals as follows:

* OSI Micro, Basic and the Personal Computer by T A Dwyer & M Critchfield.
* How to Program Microcomputers for the C3 by W Barden.
* Full set of OSI CPM manuals.
* Challenger III Set Up and Operations Manual.
* USCD System manuals.
* OS-65D and OS-65U users manuals.
* OSI-FORTH installation and user manuals with source listings.
* MS6500 Microcomputer Family Programming Manual by MOS Technology.
* Challenger II and III series hardware, circuit diagrams and servicing data manual. (and M6800 programming books).

Price : $5200 or nearest offer.
Contact J. Rankin, CCS Publications,

Registered by Australia Post
Publication No. VBG4212

If undeliverable return to
KAOS, 10 Forbes St
Essendon, Victoria 3040